Universal Re-encryption for Mixnets

Abstract. We introduce a new cryptographic technique that we call *universal re-encryption*. A conventional cryptosystem that permits re-encryption, such as ElGamal, does so only for a player with knowledge of the public key corresponding to a given ciphertext. In contrast, universal re-encryption may be performed without knowledge of public keys. We demonstrate an asymmetric cryptosystem with universal re-encryption that is half as efficient as standard ElGamal in terms of both computation and storage.

While technically and conceptually simple, universal re-encryption leads to new types of functionality in mixnet architectures. Conventional mixnets are often called upon to enable players to communicate with one another through channels that are *externally anonymous*, i.e., that hide information permitting traffic-analysis. Universal re-encryption permits a mixnet of this kind to be constructed in which servers hold *no public or private keying material*, and may therefore dispense with the cumbersome requirements of key generation, key distribution, and private-key management. We describe two practical mixnet constructions, one involving asymmetric input ciphertexts, and another with hybrid-ciphertext inputs.

Key words: anonymity, mix networks, private channels, public-key cryptography, universal re-encryption

1 Introduction

A *mix network* or *mixnet* is a cryptographic construction that invokes a set of servers to establish private communication channels [5]. One type of mix network accepts as input a collection of ciphertexts, and outputs the corresponding plaintexts in a randomly permuted order. The main privacy property desired of such a mixnet is that the permutation matching inputs to outputs should be known only to the mixnet, and no one else. In particular, an adversary should be unable to guess which input ciphertext corresponds to an output plaintext any more effectively than by guessing at random.

One common variety of mixnet known as a *re-encryption mixnet* relies on a public-key encryption scheme, such as ElGamal [11], that allows for re-encryption of ciphertexts. For a given public key, a ciphertext C' is said to represent a re-encryption of C if both ciphertexts decrypt to the same plaintext. In a re-encryption mixnet, the inputs are submitted encrypted under the public-key of the mixnet. (The corresponding private key is held in distributed form among the servers.) The batch of input ciphertexts is processed sequentially by each mix server. The first server takes the set of input ciphertexts, re-encrypts them, and outputs the re-encrypted ciphertexts in a random order. Each server in turn takes the set of ciphertexts output by the previous server, and re-encrypts and mixes them. The set of ciphertexts produced by the last server may be decrypted by a quorum of mix servers to yield plaintext outputs. Privacy in this mixnet construction derives from the fact that the ciphertext pair (C, C') is indistinguishable from a pair (C, R) for a random ciphertext R to any adversary without knowledge of the private key.

In this paper, we propose a new type of public-key cryptosystem that permits *universal re-encryption* of ciphertexts. We introduce the term universal encryption to mean re-encryption without knowledge of the public key under which a ciphertext was computed. Like standard re-encryption, universal re-encryption transforms a ciphertext C into a new ciphertext C' with

same corresponding plaintext. The novelty in our proposal is that re-encryption neither *requires* nor *yields* knowledge of the public key under which a ciphertext was computed[1].

When applied to mix networks, our universal re-encryption technique offers new and interesting functionality. Most importantly, mix networks based on universal re-encryption dispense with the cumbersome protocols that traditional mixnets require in order to establish and maintain a shared private key. We discuss more benefits and applications of universal mixnets in the next section. It is possible to construct a *universal mixnet* based on universal re-encryption roughly as follows. Every input to the mixnet is encrypted under the public key of the recipient for whom it is intended. Thus, unlike standard re-encryption mixnets, universal mixnets accept ciphertexts encrypted under the individual public keys of receivers, rather than encrypted under the unique public key of the mix network. These ciphertexts are universally re-encrypted and mixed by each server. The output of a universal mixnet is a set of ciphertexts. Recipients can retrieve from the set of output ciphertexts those addressed to them, and decrypt them.

Organization
The rest of the paper is organized as follows. In the next section, we give an overview of the main properties that distinguish universal mixnets from standard mixnets, and give one example of a new application made possible by universal mixnets. This is followed in section 3 by a formal definition of semantic security for universal re-encryption, as well as a proposal for creating a public-key cryptosystem with universal re-encryption based on ElGamal. In section 4, we describe our construction for an asymmetric universal mixnet. We define and prove the security properties of our system in section 5. In section 6, we propose a hybrid variant of our universal mixnet construction that combines public-key and symmetric encryption to handle long messages efficiently. We conclude in section 7.

2 Universal Mixnets: Properties and Applications

To motivate the constructions of this paper, we list here some of the main properties that set apart universal mixnets from traditional re-encryption mixnets. We also give one example of a new application made possible by universal mixnets: Anonymization of RFID tags.

Universal mixnets hold no keying material. A universal mixnet operates without a monolithic public key and thus dispenses at the server level with the complexities of key generation, key distribution, and key maintenance. This allows a universal mixnet to be set up more efficiently and with greater flexibility than a traditional re-encryption mixnet. A universal mixnet can be rapidly re-configured: Servers can enter and leave arbitrarily, even in the middle of a round of processing, without going through any setup. A mix server that crashes or otherwise disappears in the midst of the mixing process can thus be easily replaced by another server.

Universal mixnets guarantee forward anonymity. The absence of shared keys means that universal mixnets offer perfect forward-anonymity. Even if all mix servers become corrupted, the anonymity of previously mixed batches is preserved (provided that servers do not store the permutations or re-encryption factors they used to process their inputs). In contrast, if the keying material of a standard mix is revealed, an adversary with transcripts from previous mix sessions can compromise the privacy of users.

[1] We note that universal re-encryption has been independently devised by Danezis [7], although with a somewhat different application than we consider here.

Universal mixnets do not support escrow capability. The flip-side of perfect forward-anonymity is that is that it is not possible to escrow the privacy offered by a universal mixnet in a straightforward fashion. Escrow is only achievable in a universal mix as long as every server involved in the mixing remembers how it permuted its inputs and is willing to reveal that permutation. This may be a drawback from the perspective of law enforcement. In comparison, escrow is possible in a traditional mix, provided that the shared key can be reconstructed. This requires the participation of only a quorum of servers, not all of them.

Efficiency. We present in this paper a public-key cryptosystem with universal re-encryption that is half as efficient as standard ElGamal: It requires exactly twice as much storage, and also twice as much computation for encryption, re-encryption, and decryption. In this regard, the universal mixnet constructions we propose in this paper are practical. The drawback of a universal mixnet, as we discuss in detail below, is that receivers must attempt to decrypt all output items in order to identify the messages intended for them.

2.1 Anonymizing RFID tags

An interesting new application made possible by universal mixnets is the anonymization of radio-frequency identification (RFID) tags. An RFID tag is a small device that is used to locate and identify physical objects. RFID tags have very limited processing ability (insufficient to perform any re-encryption of data), but they allow devices to read and write to their memory [20, 21]. Communication with RFID tags is performed by means of radio, and the tags themselves often obtain power by induction. Examples of uses of RFID tags include the theft-detection tags attached to consumer items in stores and the plaques mounted on car windshields for automated toll payment. Due to the projected decrease in the cost of RFID tags, their use is likely to extend in the near future to a wide range of general consumer items, including possibly even banknotes [26, 16].

This raises concerns of an emerging privacy threat. Most RFID tags emit static identifiers. Thus, an adversary with control of a large base of readers for RFID tags may be able to track the movement of any object in which an RFID tag is embedded, and hence learn the whereabouts of the owner of that object. In order to prevent tracking of RFID tags, one could let some set of (honest-but-curious) servers perform re-encryption of the information that is publicly readable from RFID tags. The resulting system is surprisingly similar to a mix network, in which the permutation of ciphertexts is replaced by the movement of the RFID tags.

A traditional mix network, however, only partially solves the problem of tracking. The difficulty lies in the fact that the data contained in different RFID tags may be encrypted under different public keys, depending on who possesses the authority to access that data. For example, while the data contained in tags used for automated toll payment may be encrypted under the public key of the transit agency, the data contained in tags attached to merchandise in a department store may be encrypted under the public key of that department store. To re-encrypt RFID tag data, a traditional mix network would need knowledge of the key under which that data was encrypted. The public key associated with an RFID tag could be made readable, but then the public key itself becomes an identifier permitting a certain degree of tracking. This is particularly the case if a user carries a *collection* of tags, and may therefore be identified by means of a constellation of public keys.

Universal mixnets offer a means of addressing the problem of RFID-tag privacy. If the data contained in RFID tags is encrypted with a cryptosystem that permits universal re-encryption, then this data can be re-encrypted without knowledge of the public-key. Thus universal re-

encryption may offer heightened privacy in this setting by permitting agents to perform re-encryption without knowledge of public keys. While there have been previous designs using mixes for the purposes of privacy protection for low-power devices (e.g., [19]), universal re-encryption permits significant protocol and management simplification.

3 Universal Re-encryption

A conventional randomized public-key cryptosystem comprises a triple of algorithms, $CS = (\mathsf{KG}, \mathsf{E}, \mathsf{D})$, for key generation, encryption, and decryption respectively. We assume, as is often the case for discrete-log-based cryptosystems, that system parameters and underlying algebraic structures for CS are published in advance by a trusted party. These are generated according to a common security parameter k. System parameters include or imply specifications of $\mathbf{M}$, $\mathbf{C}$, and $\mathbf{R}$ – respectively a message space, ciphertext space, and set of encryption factors. In more detail:

- The key-generation algorithm $(PK, SK) \leftarrow \mathsf{KG}$ outputs a random key pair.
- The encryption algorithm $C \leftarrow \mathsf{E}(m, r, PK)$ is a deterministic algorithm that takes as input a message $m \in \mathbf{M}$, an encryption factor $r \in \mathbf{R}$ and a public key PK, and outputs a ciphertext $C \in \mathbf{C}$.
- The decryption algorithm $m \leftarrow \mathsf{D}(SK, C)$ takes as input a private key SK and ciphertext $C \in \mathbf{C}$ and outputs the corresponding plaintext.

A critical security property for providing privacy in a mix network is that of *semantic security*. Loosely speaking, this property stipulates the infeasibility of learning any information at all about a plaintext from a corresponding ciphertext [12]. For a more formal definition, we consider an adversary that is given a public key PK, where $(PK, SK) \leftarrow \mathsf{KG}$. This adversary chooses a pair (m_0, m_1) of plaintexts. Corresponding ciphertexts $(C_0, C_1) = (\mathsf{E}(m_0, r_0, PK), \mathsf{E}(m_1, r_1, PK))$ for $r_0, r_1 \in_U \mathbf{R}$ are computed, where $\in_U$ denotes uniform, random selection. For a random bit b, the adversary is given the pair (C_b, C_{1-b}), and tries to guess b. The cryptosystem CS is said to be semantically secure if the adversary can guess b with *advantage* at most negligible in k, i.e. with probability at most negligibly larger than $1/2$.

For a re-encryption mix network, an additional component known as a *re-encryption* algorithm, denoted by Re, is required in CS. This algorithm re-randomizes the encryption factor in a ciphertext. In a standard cryptosystem, this means that $C' \leftarrow \mathsf{Re}(C, r, PK)$ for $C, C' \in \mathbf{C}, r \in \mathbf{R}$, and a public key PK. Observe that re-encryption, in contrast to encryption, may be executed without knowledge of a plaintext. The notion of semantic security may be naturally extended to apply to the re-encryption operation by considering an adversary that chooses ciphertexts (C_0, C_1) under PK. The property of *semantic security under re-encryption*, then, means the following: Given respective re-encryptions (C'_b, C'_{1-b}) in a random order, the adversary cannot guess b with non-negligible advantage in k. Provided that Re yields the same distribution of ciphertexts as E (given $r \in_U \mathbf{R}$) or that the two distributions are indistinguishable, it may be seen that basic semantic security implies semantic security under re-encryption.

Bellare *et al.* [3] define another useful property possessed by the El Gamal cryptosystem. Known as "key-privacy," this property may be loosely stated as follows. Given a ciphertext encrypted under a public key randomly selected from a published pair (PK_0, PK_1), an adversary cannot determine which key corresponds to the ciphertext with non-negligible advantage. Key-privacy is one feature of the security property we develop in this paper for universal re-encryption.

As already explained, a universal cryptosystem permits re-encryption without knowledge of the public key corresponding to a given ciphertext. Let us denote such a cryptosystem by $UCS = (\mathsf{UKG}, \mathsf{UE}, \mathsf{URe}, \mathsf{UD})$, where $\mathsf{UKG}, \mathsf{UE}$, and UD are key generation, encryption, and decryption algorithms. These are defined as in a standard cryptosystem. The difference between a universal cryptosystem UCS and a standard cryptosystem resides in the re-encryption algorithm URe. The algorithm URe takes as input a ciphertext C and re-encryption factor r, but no public key PK. Thus, we have $C' \leftarrow \mathsf{URe}(C, r)$ for $C, C' \in \mathbf{C}$, $r \in \mathbf{R}$.

To define *universal semantic security under re-encryption*, i.e., with respect to URe, it is necessary to consider an adversarial experiment that is a variant on the standard one for semantic security. We define an experiment uss as follows for a (stateful) adversarial algorithm $\mathcal{A}$. This experiment terminates on issuing an output bit. As above, we assume an appropriate implicit parameterization of UCS under security parameter k. The idea behind the experiment is as follows. The adversary is permitted to construct universal ciphertexts under two randomly generated keys, PK_0 and PK_1. These ciphertexts are then re-encrypted. The aim of the adversary is to distinguish between the two re-encryptions. The adversary should be unable to do so with non-negligible advantage.

Experiment $\mathbf{Exp}^{uss}_{\mathcal{A}}(UCS, k)$
$PK_0 \leftarrow \mathsf{UKG}; PK_1 \leftarrow \mathsf{UKG};$
$(m_0, m_1, r_0, r_1) \leftarrow \mathcal{A}(PK_0, PK_1,$ "specify ciphertexts");
if $m_0, m_1 \notin \mathbf{M}$ or $r_0, r_1 \notin \mathbf{R}$ then
    output '0';
$C_0 \leftarrow \mathsf{UE}(m_0, r_0, PK_0); C_1 \leftarrow \mathsf{UE}(m_1, r_1, PK_1);$
$r'_0, r'_1 \in_U \mathbf{R};$
$C'_0 \leftarrow \mathsf{URe}(C_0, r'_0); C'_1 \leftarrow \mathsf{URe}(C_1, r'_1);$
$b \in_U \{0, 1\};$
$b' \leftarrow \mathcal{A}(C'_b, C'_{1-b},$ "guess");
if $b = b'$ then
    output '1';
else
    output '0';

We say that UCS is semantically secure under re-encryption if for any adversary $\mathcal{A}$ with resources polynomial in K, the probability $\mathsf{pr}[\mathbf{Exp}^{uss}_{\mathcal{A}}(UCS, k) = \text{'1'}] - 1/2$ is negligible in k.

The experiment uss captures the idea that the keys associated with ciphertexts are concealed by the re-encryption process in UCS. Thus, even an adversary with the opportunity to compose the ciphertexts undergoing re-encryption cannot make use of differences in public keys in order to defeat the semantic security of the cryptosystem.

3.1 Universal re-encryption based on ElGamal.

We present a public-key cryptosystem with universal re-encryption that may be based on the ElGamal cryptosystem implemented over any suitable algebraic group. The basic idea is simple: We append to a standard ElGamal ciphertext a second ciphertext on the identity element. By exploiting the algebraic homomorphism of ElGamal, we can use the second ciphertext to alter the encryption factor in the first ciphertext. As a result, we can dispense with knowledge of the public key in the re-encryption operation. As already noted, this construction is half as efficient as standard ElGamal.

Let $E[m]$ loosely denote ElGamal encryption a plaintext m (under some key). In a universal cryptosystem, a ciphertexts on message m consists of a pair $[E[m]; E[1]]$. ElGamal possesses a homomorphic property, namely that $E[a] \times E[b] = E[ab]$ for group operator $\times$. Thanks to this property, the second component can be used to re-encrypt the first without knowledge of the associated public key. To provide more detail, let $\mathcal{G}$ denote the underlying group for the ElGamal cryptosystem; let q denote the order of $\mathcal{G}$. (Here the security parameter k is implicit in the choice of $\mathcal{G}$.) Let g be a published generator for $\mathcal{G}$. The universal cryptosystem is as follows. Note that we assume random selection of encryption and re-encryption factors in this description.

- **Key generation** (UKG): Output $(PK, SK) = (y = g^x, x)$ for $x \in_U \mathcal{Z}_q$.
- **Encryption** (UE): Input comprises a message m, a public key y, and a random encryption factor $r = (k_0, k_1) \in Z_q^2$. The output is a ciphertext $C = [(\alpha_0, \beta_0); (\alpha_1, \beta_1)] = [(my^{k_0}, g^{k_0}); (y^{k_1}, g^{k_1})]$. We write $C = \mathsf{UE}_{PK}(m, r)$ or $C = \mathsf{UE}_{PK}(m)$ for brevity.
- **Decryption** (UD): Input is a ciphertext $C = [(\alpha_0, \beta_0); (\alpha_1, \beta_1)]$ under public key y. Verify $\alpha_0, \beta_0, \alpha_1, \beta_1 \in \mathcal{G}$; if not, the decryption fails, and a special symbol $\perp$ is output. Compute $m_0 = \alpha_0/\beta_0^x$ and $m_1 = \alpha_1/\beta_1^x$. If $m_1 = 1$, then the output is $m = m_0$. Otherwise, the decryption fails, and a special symbol $\perp$ is output. Note that this ensures a binding between ciphertexts and keys: a given ciphertext can be decrypted only under one given key.
- **Re-encryption** (URe): Input is a ciphertext $C = [(\alpha_0, \beta_0); (\alpha_1, \beta_1)]$ with a random re-encryption factor $r' = (k'_0, k'_1) \in Z_q^2$. Output is a ciphertext $C' = [(\alpha'_0, \beta'_0); (\alpha'_1, \beta'_1)] = [(\alpha_0\alpha_1^{k'_0}, \beta_0\beta_1^{k'_0}); (\alpha_1^{k'_1}, \beta_1^{k'_1})]$, where $k'_0, k'_1 \in_U \mathcal{Z}_q$.

Observe that the ciphertext size and the computational costs for all algorithms are exactly twice those of the basic ElGamal cryptosystem. The properties of standard semantic security and also universal semantic security under re-encryption (as characterized by experiment uss) may be shown straightforwardly to be reducible to the Decision Diffie-Hellman (DDH) assumption [4] over the group $\mathcal{G}$, in much the same way as the semantic security of ElGamal [25]. Thus, one possible choice of $\mathcal{G}$ is the subgroup of order q of $\mathcal{Z}_p^*$, where p and q are primes such that $q \mid p - 1$. An alternative, with the advantage of more compact ciphertext representation, is a group of prime order q defined over an appropriately selected elliptic curve such that the DDH assumption is believed to be hard. Throughout the remainder of the paper, we work with the ElGamal implementation of universal re-encryption, and let g denote a published generator for the choice of underlying group $\mathcal{G}$.

4 Universal Mix Network Construction

We use the following scenario to introduce our universal mixnet construction. We consider a number of senders who wish to send messages to recipients in such a way that the communication is concealed from everyone but the sender and recipient themselves. In other words, we wish to establish channels between senders and receivers that are *externally anonymous*. We assume that every recipient has an ElGamal private/public key pair $(x, y = g^x)$ in some published group $\mathcal{G}$. We also assume that every sender knows the public key of all the receivers with whom she intends to communicate. (Alternatively, the sender may have a "blank" ciphertext for this party. By this we mean an encryption using UE of the identity element in $\mathcal{G}$ under the public key of the recipient. A "blank" may be filled in without knowledge of the corresponding public key through exploitation of the underlying algebraic homomorphism in ElGamal.) The communication protocol proceeds as follows:

1. **Submission of inputs.** Senders post to a bulletin board messages that are universally encrypted under the public key of the recipient for whom they are intended. Every entry on the bulletin board thus consists of a pair of ElGamal ciphertexts $(E[m]; E[1])$ under the public key of the recipient. Recall that the semantic security of ElGamal ensures the concealment of plaintexts. In other words, for plaintexts m and m', a universal ciphertext $(E[m]; E[1])$ is indistinguishable from another $(E[m']; E[1])$ to any entity without knowledge of the corresponding private key.
2. **Universal mixing.** Any server can be called upon to mix the contents of the bulletin board. This involves two operations: (1) The server re-encrypts all the universal ciphertexts on the bulletin board using URe, and (2) The server writes the resulting new ciphertexts back to the bulletin board in random order, overwriting the old ones. It is also desirable that a server that mixes the inputs be able to prove that it operated correctly. This can be done using a number of existing mixing schemes, e.g. [1, 2, 10, 13, 15, 17], and will be discussed in greater detail below.
3. **Retrieval of the outputs.** Potential recipients must try to decrypt every encrypted message output by the universal mixnet. Successful decryptions correspond to messages that were intended for that recipient. The others (corresponding to decryption output '$\perp$') are discarded by the party attempting to perform the decryption. Recall that our construction of universal encryption based on El Gamal ensures a binding between ciphertexts and keys, so that a given ciphertext can be decrypted only under one given key.

Properties of the basic protocol:

1. The universal mixnet holds no keying information. Public and private keys are managed exclusively by the players providing input ciphertexts and receiving outputs from the mix.
2. The universal mixnet guarantees only external anonymity. It does not provide anonymity for senders with respect to receivers. Indeed a receiver can trace a message intended for her throughout the mixing process, since that message is encrypted under her public key. If ciphertexts are not posted anonymously, this means that the receiver can identify the players who have posted messages for her. This restriction to external anonymity is of little consequence for the applications we focus on, namely protection against traffic analysis, but should be borne in mind for other applications.
3. The chief drawback of universal mixnets is the overhead that they impose on receivers. Because the public keys corresponding to individual output ciphertexts are unknown, it may be necessary for a receiver to attempt to decrypt each output ciphertext in order to find the right one, i.e., the ciphertext corresponding to her private key. Thus, a universal mixnet imposes an overhead on receivers that is linear in the input batch size. (We discuss ways below and in section 6 to reduce this overhead somewhat.)

Low-volume anonymous messaging: anonymizing bulletin boards.
For simplicity, we have described above the operation of a universal mixnet in which inputs are submitted, mixed and finally retrieved. This sequence of events is characteristic of all mixes. Unlike regular mixes however, universal mixes allow for repeated interleaving of the submission, mixing and retrieval steps. What makes this possible is that the decryption is performed by the recipients of the message rather than by the mixnet, so that existing messages posted to the bulletin board are at all times indistinguishable from new messages. New inputs may be constantly added to the existing content of the bulletin board, and outputs retrieved, provided there is at least one round of mixing between every submission and retrieval to ensure privacy.

This suggests a generalization of the private communication protocol described above, in which the bulletin board maintains at all times a pool of unclaimed messages. In other words,

universal mixing lends itself naturally to the construction of an *anonymizing bulletin board*. Senders may add messages and receivers retrieve them at any time, provided there is always at least one round of mixing between each posting and retrieval. This protocol appears well suited to guarantee anonymity from external observers in a system in which few messages are exchanged. The privacy of the protocol relies on the existence of a steady pool of undelivered messages rather than on a constant flow of new messages. The former condition appears much easier to satisfy than the latter in cases when the total number of exchanged messages is small. This pooling of messages affords good anonymity protection, without the usual lack of verifiability of correct performance that vexes such schemes.[2]

A potential drawback of a bulletin board based on universal mixing is that one must download the full contents in order to be assured of obtaining all of the messages addressed to oneself. This becomes problematic if the number of messages on the bulletin board is permitted to grow indefinitely. To mitigate this problem, it is possible to have recipients remove the messages they have received.[3] An anonymizing bulletin board based on universal mixing has the important privacy-protecting feature that removal of a particular message does not reveal which entity posted that message. Another important observation, as described in the next section, is that only a portion of each message on a bulletin board need be downloaded to allow a recipient to determine which messages are intended for her. This further restricts the work required by a receiver.

RFID-tag privacy.
Universal re-encryption may be used to enhance the privacy of RFID tags. The idea is to permit powerful computing agents external to RFID tags to universally re-encrypt the tag data (recall that the tags lack the computing power necessary to do the re-encryption themselves). Thus, for example, a consumer walking home with a bag of groceries containing RFID tags might have the ciphertexts on these tags re-encrypted by computing agents that are provided as a public service by shops and banks along the way. In this case, the tags in the bag of groceries will periodically change appearance, helping to defeat any tracking attempt.

Application of universal mixnets to RFID-tag privacy is different in some important respects from realization of an anonymous bulletin board. As re-encryption naturally occurs for RFID tags on an individual basis, re-encryption in this setting may be regarded as realizing an *asynchronous* mixnet. There is also a special security consideration in this setting. Suppose that the ciphertext on an RFID tag is of the form $(\alpha, \beta); (1, 1)$ (where '1' represents the identity element for $\mathcal{G}$). Then the ciphertext on the tag will not change upon re-encryption. Thus, it is important to prevent an active adversary from inserting such a ciphertext onto an RFID tag so as to be able to trace it and undermine the privacy of the possessor. In particular, on processing ciphertexts, re-encryption agents should check that they do not possess this degenerate form. Of course, an adversary in this environment can always corrupt ciphertexts. Note, however, that even a corrupted ciphertext $(\alpha', \beta'); (\gamma, \delta)$ will be rendered unrecognizable to an adversary provided that $\gamma, \delta \neq 1$.

[2] So-called pool mixes typically use processing delays in asynchronous settings to hide timing information. They were first described by Lance Cottrell in the nineties [6]. See [23] for a further discussion of pool mixes, and [9] for an approach to verifying correct functioning of pool mixes.

[3] To ensure that messages are only removed by the intended recipient, a proof of knowledge of the corresponding decryption key is required. Note that such a proof can be performed without disclosing the public key associated with the required decryption key. For ciphertext $C = [(\alpha_0, \beta_0); (\alpha_1, \beta_1)]$, this may take the form of a non-interactive zero-knowledge proof of knowledge of an exponent x such that $\alpha_1 = \beta_1^x$ – essentially a Schnorr signature [22].

5 Security

In this section, we define two security properties of universal mixnets:

- **Correctness:** The mixnet is correct if the set of output it produces is a permutation of the set of inputs.
- **Communication privacy:** The mixnet guarantees communication privacy if, when Alice sends a message to Bob and Cathy sends a message to Dario, an observer can not tell whether Alice (resp. Cathy) sent a message to Bob or Dario.

Correctness. Correctness for universal mixnets follows directly from the definition of correctness for standard mixnets. Like standard mix servers, universal servers must prove that they have performed the mixing operation correctly. For this, it is possible to draw on essentially any of the proof techniques presented in the literature on mixnets, as nearly all apply to ElGamal ciphertexts. For example, to achieve universal verifiability, it is possible to employ the proof techniques in [10, 17, 15]. A small technical consideration, which may be dealt with straightforwardly, is the form of input ciphertexts. Input ciphertexts in most mix network constructions consist of a single ElGamal ciphertext, while in our construction, an input consists of a universal ciphertext, and thus two related ElGamal ciphertexts.

Communication privacy. We define next the property of communication privacy. In order to state this definition formally, we abstract away some of the operations of the mixnet by defining them in terms of oracle operations. We do this so as to focus our exposition on our universal construction, rather than underlying primitives, particularly as our construction can make use of a broad range of choices of such primitives. We define three oracles:

- **An oracle** `MIX` It universally re-encrypts all ciphertexts on the bulletin board BB and outputs back to BB the new set of ciphertexts in a randomly permuted order. In practice, any mix network with public verifiability may be substituted for our oracle `MIX`.
- **An oracle** `POST` that permits message posting. This oracle requires a poster to submit a message, encryption factors and ciphertext. The oracle verifies that the message, encryption factors and ciphertext are elements of the appropriate groups. The oracle permits posting if the ciphertext is a valid encryption of the message with the given encryption factors. Note that the oracle `POST` may be regarded as simulating a proof of knowledge of the plaintext and the encryption factor and a verification thereof. In practice, it could be instantiated with standard discrete-log-based proofs of knowledge, e.g., [8], in either their interactive or non-interactive forms.
- **An oracle** `RETRIEVE` that permits message retrieval. The oracle takes a private key and ciphertext from a user. The oracle verifies that the private key and ciphertext are elements of the appropriate groups. The user is allowed to remove the ciphertext if it is encrypted under the private key. Recall that our construction of universal encryption based on El Gamal ensures a binding between ciphertexts and keys, so that a given ciphertext can be decrypted only under one given key. The oracle `RETRIEVE`, like `POST`, abstracts away a proof of knowledge of the plaintext.

We define communication privacy in terms of an experiment $\mathbf{Exp}^{comm-priv}$ defined as follows. The adversary may make an arbitrary number of calls to any of the oracles `RETRIEVE`, `MIX`, or `POST` and may order these calls as desired. We enumerate the first several steps here for reference in our proof.

Experiment $\mathbf{Exp}_{\mathcal{A}}^{comm-priv}(UCS, k)$

1. $PK_0 \leftarrow \mathsf{UKG}; PK_1 \leftarrow \mathsf{UKG}$;
2. $(m_0, m_1) \leftarrow \mathcal{A}(PK_0, PK_1,$ "specify plaintexts");
3. $b \in_U \{0, 1\}$;
4. $C'_0 = \mathsf{UE}_{PK_b}(m_b)$ and $C'_1 = \mathsf{UE}_{PK_{1-b}}(m_{1-b})$ appended to BB;
5. `MIX` invoked;
6. $\mathcal{A}(BB)$;
7. $L \leftarrow \{C \in BB$ s.t. C is a valid ciphertext under $PK_0\}$;
8. $b' \leftarrow \mathcal{A}(L,$ "guess b");

if $b = b'$ then
    output '1';
else
    output '0';

An intuitive description of this experiment is as follows. Alice and Bob wish each to transmit a single message to one of Cathy and Dario, who possess public keys PK_0 and PK_1 respectively. Our aim is to ensure that the adversary cannot tell whether Alice is sending a message to Cathy or Dario – and likewise to whom Bob is transmitting. The adversary is given the special (strong) power of determining which plaintexts, m_0 and m_1, are to be received by Cathy and Dario. The adversary observes Alice posting ciphertext C'_0 and Bob posting ciphertext C'_1, but does not know which ciphertext is for Cathy and which is for Dario. The bulletin board is then subjected to a mixing operation so as to conceal the communication pattern. The adversary may subsequently control when and how the mix network is invoked, and may place its own ciphertexts on the bulletin board. Finally, at the end of the experiment, the adversary is given a list L of all ciphertexts encrypted under PK_0, i.e., all the messages that Cathy retrieves. This list L will include the one such message posted by Alice or Bob in addition to all messages encrypted under PK_0 and posted by the adversary. The task of the adversary is to guess whether it was Alice who sent a message to Cathy (case $b = 0$) or Bob (case $b = 1$).

Definition 1. (Communication privacy) *We say that a universal mixnet for UCS possesses* communication privacy *if for any adversary $\mathcal{A}$ that is polynomial time in k, we have $\mathsf{pr}[\mathbf{Exp}_{\mathcal{A}}^{comm-priv}(UCS, k) = 1] - 1/2$ is negligible in k.*

Theorem 1. *Our universal mixnet possesses communication privacy provided that UCS has universal semantic security under re-encryption. For our described construction involving ElGamal, privacy may consequently be reduced to the DDH assumption over $\mathcal{G}$.*

Proof: Assume that we have an adversary $\mathcal{A}$ for which $\mathsf{pr}[\mathbf{Exp}_{\mathcal{A}}^{comm-priv}(UCS, k) = 1] - 1/2$ is non-negligible in k. We build a new adversary $\mathcal{A}'$ which uses $\mathcal{A}$ as a subroutine and for which $\mathsf{pr}[\mathbf{Exp}_{\mathcal{A}'}^{uss}(UCS, k) =$ '1'$] - 1/2$ is non-negligible in k (i.e. $\mathcal{A}'$ breaks the universal semantic security of the underlying encryption scheme). $\mathcal{A}'$ operates as follows:

- At the beginning of the experiment $\mathbf{Exp}^{uss}$, $\mathcal{A}'$ is given two public keys PK_0 and PK_1. $\mathcal{A}'$ gives these two keys to $\mathcal{A}$. This simulates step 1 of $\mathbf{Exp}^{comm-priv}$.
- When $\mathcal{A}$ calls one of the oracles `POST`, `MIX` or `RETRIEVE`, $\mathcal{A}'$ can trivially simulate the oracle for the requested operation for $\mathcal{A}$.
- In step 2 of experiment $\mathbf{Exp}^{comm-priv}$, $\mathcal{A}$ specifies plaintexts m_0 and m_1. $\mathcal{A}'$ selects random encryption factors r_0 and r_1 and computes $C_0 = \mathsf{UE}_{PK_0}(m_0, r_0)$ and $C_1 = \mathsf{UE}_{PK_1}(m_1, r_1)$. $\mathcal{A}'$ submits these in the second step of experiment $\mathbf{Exp}^{uss}$. $\mathcal{A}'$ then receives as input from experiment $\mathbf{Exp}^{uss}$ two new ciphertexts C'_0 and C'_1.

– In step 4 of $\mathbf{Exp}^{comm-priv}$, $\mathcal{A}'$ posts C'_0 and C'_1 to the bulletin board.
– In step 7 of $\mathbf{Exp}^{comm-priv}$, $\mathcal{A}'$ must identify the set of outputs encrypted under PK_0. Note that $\mathcal{A}'$ can easily identify among the outputs that correspond to inputs originally submitted by $\mathcal{A}$ those encrypted under PK_0, since it controls the oracle `POST` and `MIX`. The only difficulty is for $\mathcal{A}'$ to decide which of C'_0 and C'_1 is encrypted under PK_0 and which under PK_1. Since $\mathcal{A}'$ doesn't know that, it arbitrarily assigns C'_0 to the list L of ciphertexts encrypted under PK_0.

In the last step of the simulation, $\mathcal{A}'$ assigns C'_0 arbitrarily to L. We claim that if $\mathcal{A}$ can distinguish between the case where this assignment to L is correct and the case where it is incorrect, then $\mathcal{A}$ can be used to break universal semantic security in $\mathbf{Exp}^{uss}$. This may be achieved with a small modification of our simulation as follows: (1) $\mathcal{A}'$ lets $C'_0 = C_0$ and $C'_1 = C_1$, but invokes $\mathbf{Exp}^{uss}$ on the pair (C'_0, C'_1) during the mixing operation in step 5 and (2) $\mathcal{A}'$ submits to Exp^{uss} the bit b' yielded by $\mathcal{A}$ at the end of the experiment. Let us assume, therefore, that the assignment to L is correct.

Given this, when $\mathcal{A}$ outputs its guess b', $\mathcal{A}'$ then outputs the same bit b' as its guess for the experiment Exp^{uss}. It is clear now that when $\mathcal{A}$ guesses correctly, so does $\mathcal{A}'$. This concludes our proof. □

Security of UCS and chosen-ciphertext attacks.
The cryptosystem UCS we employ here inherits the semantic security property of the underlying El Gamal cipher under the DDH assumption. This property is critical to our definition of communications privacy. Our model for communications privacy makes one simplifying assumption that must be noted, though: We assume that the adversary does not learn any information about plaintexts. For this reason, we do not require adaptive-chosen ciphertext (CCA) security of our cryptosystem. In fact, we cannot achieve CCA security in the strictest sense in our system: In order to permit re-encryption, ciphertext must be malleable. Note, however, that because of the need to demonstrate knowledge of the plaintext and encryption factors in the `POST` operation, it is infeasible for an adversary to re-post a message or to post a new message with a related plaintext.

On the other hand, there may be circumstances in which an adversary may indeed learn information about plaintexts in our system. To show this in a formal sense, however, it would be necessary to modify our universal cryptosystem so as to achieve CCA security with *benign malleability*, as defined by Shoup [24]. In Shoup's terminology, we would need to require an induced *compatible relation* of plaintext equivalence by formatting plaintexts with appropriate padding. We omit detailed discussion of this topic, however, in this paper. An adversary that can gain significant information about received messages can, after all, break the basic privacy guarantees of the system.

6 Hybrid universal mixing

We describe next a variant mixnet called a *hybrid universal mixnet*. This type of mixnet combines symmetric and public-key encryption to accommodate potentially very long messages (all of the same size) in an efficient manner. We refer the interested reader to [18, 14] for definitions and examples of hybrid mixnets. Our definition of a universal hybrid mix considers a weaker threat model than above with respect to correctness. Our universal hybrid mix cannot be verified to correctly execute the protocol because of the use we make of symmetric encryption.

Thus, we restrict our security model to mix servers subject only to passive adversarial corruption. Such servers are also known as *honest-but-curious*. They follow the protocol correctly but try to learn as much information as possible from its execution.

For efficiency, inputs m are submitted to a hybrid mix encrypted under an initial symmetric (rather than public) key. We denote by $\epsilon_k[m]$ the symmetric-key encryption of m under key k. Each mix server S_i consecutively re-encrypts the output of the previous mix under a new random symmetric key k_i. If there are k mix servers, the final output of the mix is therefore $\epsilon_{k_n}[\epsilon_{k_{n-1}}[\ldots\epsilon_{k_1}[\epsilon_k[m]]\ldots]$. The symmetric keys $k, k_1, \ldots, k_n$ must be conveyed alongside the encrypted message to enable decryption by the final recipient. These keys are themselves encrypted as universal ciphertexts under the public key of the recipient. Universal encryption provides a very efficient way of transmitting encryptions of the symmetric keys in a way that does not compromise privacy.

Let us now give a more detailed definition of our hybrid universal mixnet. Our construction imposes an upper bound n on the maximum number of times that the mixing operation is performed by the mixnet on any given ciphertext. The protocol consists of the following steps:

1. **Submission of inputs.** An input ciphertext takes the form

$$\epsilon_{k_0}[m], E[1], (E[k_0], E[1] \ldots E[1])$$

   where $\epsilon_{k_0}[m]$ denotes symmetric-key encryption of m under key k_0. This is followed by an encryption of 1, and by a vector of ciphertexts on keys, where only the first element is filled in (with k_0), leaving the remaining $n-1$ elements as encryptions of 1.
2. **Universal mixing.** The i^{th} server to perform the mixing operation does the following for each of the ciphertexts on the bulletin board:
   - Generates a random symmetric key k_i;
   - Adds a new layer of symmetric encryption to m under key k_i;
   - Uses the second element, $E[1]$, to compute an encryption of k_i – call this $E[k_i]$;
   - Rotates the elements of the vector one step leftwards, then substituting the first element with $E[k_i]$; and
   - Re-encrypts the second element and each element of the vector.

   When it has thus processed all its inputs in this manner, the server outputs them back to the bulletin board in a random order.
3. **Retrieval of the outputs.** At the end of $d \leq n$ mixing operations, the final output of the mixnet assumes the form:

$$\epsilon_{k_d}[\epsilon_{k_{d-1}}[\ldots\epsilon_{k_0}[m]]\ldots], E[1], (\{E[1]\}^{n-d}, E[k_0] \ldots E[k_d]),$$

   where $\{E[1]\}^{n-d}$ denotes $n-d$ ElGamal ciphertexts on the identity element. As before, recipients try to decrypt every output of the mixnet and discard those outputs for which the decryption fails. Only the second element, $E[1]$, however, has to be decrypted in order for a party to determine whether the ciphertext is intended for her.

Remark: In principle, it is possible to use the "blank" ciphertext $E[1]$ to append ciphertexts on as many symmetric keys as desired, and thus re-encrypt indefinitely. The reason for restricting the number of "blank" ciphertexts to exactly n is to preserve a uniform length, without which an adversary can distinguish among ciphertexts that have undergone differring numbers of re-encryptions. A drawback of this approach is that a ciphertext re-encrypted more than n times will become undecipherable by the receiver. Given enough messages, it is alternatively possible to permit messages to grow in sizes according to their "ages", i.e., the number of re-encryptions they have undergone, and to pool them accordingly.

7 Conclusion

Universal re-encryption represents a simple modification to the basic El Gamal cryptosystem that permits re-randomization of ciphertexts without knowledge of the corresponding private key. This provides a valuable tool, as we show, for the construction of privacy-preserving architectures that dispense with the complications and risks of distributed key setup and management. The costs for the basic universal cryptosystem are only twice those of ordinary El Gamal. On the other hand, the problem of receiver costs in a universal mixnet presents a compelling line of further research. In the construction we have proposed, a receiver must perform a linear number of decryptions to identify messages intended for her. A method for reducing this cost would be appealing from both a technical and practical standpoint.

References

1. M. Abe. Mix-networks on permutation networks. In K-Y. Lam, E. Okamoto, and C. Xing, editors, *ASIACRYPT '99*, volume 1716 of *Lecture Notes in Computer Science*, pages 258–273. Springer-Verlag, 1999.
2. M. Abe and F. Hoshino. Remarks on mix-networks based on permutation networks. In *PKC '01*, pages 317–324. Springer-Verlag, 2001. LNCS no. 1992.
3. M. Bellare, A. Boldreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In C. Boyd, editor, *ASIACRYPT '01*, pages 566–582, 2001. LNCS no. 2248.
4. D. Boneh. The Decision Diffie-Hellman problem. In *ANTS '98*, pages 48–63. Springer-Verlag, 1998. LNCS no. 1423.
5. D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.
6. L. Cottrell. Mixmaster & remailer attacks, 1995. http://www.obscura.com/ loki/ remailer/remailer-essay.html.
7. G. Danezis, 2002. Personal communication.
8. A. de Santis, G. di Crescenzo, G. Persiano, and M. Yung. On monotone formula closure of SZK. In *FOCS '94*, pages 454–465. IEEE Press, 1994.
9. E. Franz, A. Graubner, A. Jerichow, and A. Pfitzmann. Comparison of commitment schemes used in mix-mediated anonymous communication for preventing pool-mode attacks. In C. Boyd and E. Dawson, editors, *ACISP '98*, pages 111–122. Springer-Verlag, 1998. LNCS no. 1438.
10. J. Furukawa and K. Sako. An efficient scheme for proving a shuffle. In J. Kilian, editor, *CRYPTO '01*, volume 2139 of *Lecture Notes in Computer Science*, pages 368–387. Springer-Verlag, 2001.
11. T. El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, 1985.
12. S. Goldwasser and S. Micali. Probabilistic encryption. *J. Comp. Sys. Sci*, 28(1):270–299, 1984.
13. M. Jakobsson and A. Juels. Millimix: Mixing in small batches, June 1999. DIMACS Technical Report 99-33.
14. M. Jakobsson and A. Juels. An optimally robust hybrid mix network. In *PODC '01*, pages 284–292. ACM Press, 2001.
15. M. Jakobsson, A. Juels, and R. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In D. Boneh, editor, *USENIX '02*, pages 339–353, 2002.
16. A. Juels and R. Pappu. Squealing euros: Privacy protection in rfid-enabled banknotes. In R. Wright, editor, *Financial Cryptography 2003*, 2003. To appear.
17. A. Neff. A verifiable secret shuffle and its application to e-voting. In P. Samarati, editor, *ACM CCS '01*, pages 116–125. ACM Press, 2001.
18. M. Ohkubo and M. Abe. A length-invariant hybrid mix. In T. Okamoto, editor, *ASIACRYPT '00*, volume 1976 of *Lecture Notes in Computer Science*, pages 178–191. Springer-Verlag, 2000.
19. M. Reed, P. Syverson, and D. Goldschlag. Protocols using anonymous connections: mobile applications. In *Security Protocols '97*, pages 13–23. Springer-Verlag, 1997. LNCS 1361.
20. S. Sarma. Towards the five-cent tag. Technical Report MIT-AUTOID-WH-006, MIT Auto ID Center, 2001. Available from http://www.autoidcenter.org/.
21. S. Sarma. Radio-frequency identification systems. In B. Kaliski, editor, *CHES '02*. Springer-Verlag, 2002. To appear.
22. C.-P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.

23. A. Serjantov, R. Dingledine, and P. Syverson. From a trickle to a flood: active attacks on several mix types. In *Information Hiding '02*, pages 36–52. Springer-Verlag, 2002. LNCS no. 2578.
24. V. Shoup. A proposal for an iso standard for public key encryption (version 2.1), 20 December 2001. Manuscript.
25. Y. Tsiounis and M. Yung. On the security of ElGamal-based encryption. In *Workshop on Practice and Theory in Public Key Cryptography (PKC '98)*, pages 117–134. Springer, 1998. LNCS no. 1431.
26. J. Yoshida. Euro bank notes to embed RFID chips by 2005. *EE Times*, 19 December 2001. Available at http://www.eetimes.com/story/OEG20011219S0016.